EXHIBIT 229

CASTLE HILL GAMING

Release Notes for CHG Class 2 Gaming System
10 May 2018 Submission

Preface

Important Note: with this submission, the CHG Class II Gaming System has returned to its original scheme used for bingo card generation, identification, and seeding. Specifically, bingo cards are identified and seeded using a GUID value.

In order to ensure Bingo Card uniqueness using such a GUID-based scheme, we have added functionality on the Game Server for storing information on which bingo cards have been played against each active game/ball-call. With each new game-play request received by the Game Server, the bingo card associated with the incoming game-play request is compared to all prior bingo cards played against the associated game/ball-call. If a duplicate bingo card were to be found as a result of this comparison, the game-play request would be disallowed.

Consequently, all source code and configuration relating to the uint32-based bingo card generation, identification, and seeding introduced in prior submissions has been removed. This includes the concepts of master deck, EGM decks, uint32-based bingo card seeding, replaceable vs. non-replaceable card decks and all other related concepts.

For purposes of file differences included in this release, the points of reference are noted in the specific sections which follow.

Games 25.2.0:

1. Removed `<bingoCardDeckType>` configuration option from all gameConfig.xml files.
2. Point of reference for file differences: Games 25.1.0 (from 15-Nov-2017 submission). Note however that there are no source file differences relative to this reference. The only difference is the configuration file change noted above.

Game Platform 6.0.7.4:

1. Updates/Reversions for linking to shared client-server libraries which use the GUID-based bingo card generation, identification, and seeding scheme with new card uniqueness verification.
2. Point of reference for file differences: Platform 6.0.7.1 (from 18-Aug-2017 submission).

Game Client 3.2.0:

1. Updates/Reversions for using the GUID-based bingo card generation, identification, and seeding scheme with new card uniqueness verification. Refer to Appendix A below for details.
2. Point of reference for file differences: Game Client 3.0.0 (from 18-Aug-2017 submission).

Castle Hill Gaming, LLC
1807 Seminole Trail, Suite 204; Charlottesville, VA 22901
Phone: 434-CHG-7777
www.castlehillgaming.com

CASTLE HILL GAMING

Game Server 2.2.0:

1. Updates/Reversions for using the GUID-based bingo card generation, identification, and seeding scheme with new card uniqueness verification. Refer to Appendix A below for details.
2. Introduction of 'UsedBingoCardFaces' table to data store which maintains list of all bingo cards played against each game/ball-call. Refer to Appendix A below for details.
3. Point of reference for file differences: Game Server 2.0.0 (from 18-Aug-2017 submission).

Game Server Emulator 2.2.0:

1. Updates/Reversions for using the GUID-based bingo card generation, identification, and seeding scheme with new card uniqueness verification. Refer to Appendix A below for details.
2. Point of reference for file differences: Game Server Emulator 2.0.0 (from 18-Aug-2017 submission).

Appendix A – GUID-based Bingo Card Scheme with Uniqueness Verification:

With this submission, the CHG Class II Gaming System has returned to its original GUID-based scheme used for bingo card generation, identification, and seeding. And, in order to ensure that no bingo card will be utilized multiple times for game-play against a single ball-call, a bingo card uniqueness verification check has been added.

As per the original GUID-based scheme, bingo card id's/seeds are generated by the Game Client (using the standard Microsoft .NET GUID generation API). With such GUID's, bingo card face values are generated in a deterministic scheme by using the GUID card id/seed to seed a random number generator. Then values are deterministically pulled from the seeded RNG to populate the face values of the bingo card.

When a new game-play request is received by the Game Server, the Game Server now checks for uniqueness of the card cell values of the bingo card sent with the incoming game-play request. For this, the Game Server maintains a table in its data store (named 'UsedBingoCardFaces') and checks that the incoming bingo card's cell values are unique relative to those of all other bingo cards played against the specific game/ball-call being used to satisfy the game-play request. If the uniqueness check were to fail, the game-play response would indicate such, and, consequently, the game-play would be aborted on the client.

Additional details of the uniqueness check are as follows:

1. Upon receiving an incoming game-play request, the Game Server generates the bingo card cell values using bingo card GUID id/seed included in the request message.
2. The Game Server then converts each of the bingo card's 25 integer cell values into 4-bit values (with appropriate offsets for the 'INGO' column cell values) and packs such bits into the first 13 bytes (104 bits) of a 16-byte (128-bit) GUID. Note that the GUID corresponding to the packed cell values is distinct from (but related to / derivable from) the card's GUID id/seed.

CASTLE HILL GAMING

3. The Game Server then attempts to insert a record into the 'UsedBingoCardFaces' data table containing the packed cell GUID value generated in Step 2 above along with the ID (also a GUID) of the game/ball-call which has been selected to satisfy the game-play request. Note that the primary key for the 'UsedBingoCardFaces' is defined by the composite value of the game/ball-call GUID table column and the packed cell values GUID table column. Thus, the database will reject the record insertion attempt if the PK uniqueness constraint is violated. In such cases, a DbUpdateException will be thrown from the Entity Framework layer and handled appropriately by the Game Server such that the game-play response relays a message to the client to abort/disallow the game-play attempt with the specified bingo card id/seed.

Note that when a game/ball-call is ended (via achievement of the game-end win by a game client), all records in the 'UsedBingoCardFaces' table corresponding to the ended game/ball-call will be deleted.

For details of this new GUID-based bingo card implementation, refer to the source code in the following files:

- `CastleHill.SharedClientServer.Games.Bingo.BingoCard.cs`
- `CastleHill.GameServer.Engine.Gamee.Bingo.BingoCardFace.cs`
- `CastleHill.GameServer.Engine.DataAccessLayer.Config.BingoCardFaceConfiguration.cs`
- `CastleHill.GameServer.Engine.DataAccessLayer.DAO.BingoCardFaceDao.cs`
- `CastleHill.GameServer.Engine.Egms.BingoEgm.cs`
- `CastleHill.GameServer.Engine.Games.Bingo.BingoGameBase.cs`
- `CastleHill.GameServer.Engine.DataAccessLayer.DbContext.ChgGameServerDbContext.cs`
- `CastleHill.Utilities.DataConsumedEventArgs.cs`
- `CastleHill.GameServer.Engine.Core.GameFloor.cs`
- `CastleHill.Utilities.GameId.cs`
- `CastleHill.Interfaces.Server.Game.Bingo.IBingoCard.cs`
- `CastleHill.Interfaces.Server.Game.Bingo.IBingoCardFace.cs`
- `CastleHill.Interfaces.Server.DataAccess.IBingoCardFaceDao.cs`
- `CastleHill.Interfaces.Server.Game.Bingo.IBingoCardPacker.cs`
- `CastleHill.Interfaces.Server.Game.IGameId.cs`
- `CastleHill.Interfaces.Shared.IStartable.cs`
- `CastleHill.Utilities.ProduceConsume.cs`
- `CastleHill.SharedClientServer.Games.Bingo.StandardBingoCardPacker.cs`